

Color Detection using Pandas & OpenCV

A MINI PROJECT REPORT

Submitted to University of Mumbai

In

COMPUTER ENGINEERING

Submitted By

SHREERAM GEEDH

VU1S1819070

ATHARVA HANJANKAR

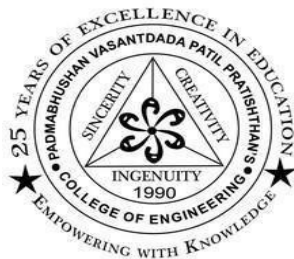
VU1S1819064

SELVAGANAPATHI MURUGESAN

VU1S1819054

Under the Guidance Of

SUMIT SHINDE



DEPARTMENT OF COMPUTER ENGINEERING
PADMABHUSHAN VASANTDADA PATIL PRATISHTHAN'S
COLLEGE OF ENGINEERING

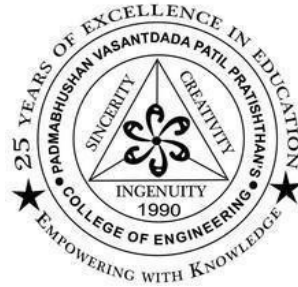
SION, MUMBAI-400022

UNIVERSITY OF MUMBAI

2019-2020

PADMABHUSHAN VASANTDADA PATIL PRATISHTHAN'S COLLEGE OF
ENGINEERING

SION, MUMBAI- 400022



CERTIFICATE

This is to certify that the mini project work, entitled has **Color Detection**
usingPandas & OpenCV completed successfully in Computer
Engineering by

Class: TE

Division: B

NAME	VID
Shreeram Geedh	VU1S1819070
Atharva Hanjankar	VU1S1819064
Selvaganapathi Murugesan	VU1S1819054

PADMABHUSHAN VASANTDADA PATIL PRATISHTHAN'S COLLEGE OF
ENGINEERING

SION, MUMBAI- 400022



MINI PROJECT APPROVAL CERTIFICATE

This is to certify that the mini project work of semester 6, entitled Color Detection using Pandas & OpenCV is approved in Computer Engineering completed by

NAME	VID
Shreeram Geedh	VU1S1819070
Atharva Hanjankar	VU1S1819064
Selvaganapathi Murugesan	VU1S1819054

Internal Examiner

External Examiner

Name

Name

Date

Date

Mini project Convener

Head of Department

Principal

ABSTRACT

OpenCV is a Computer Vision library. It is a collection of C functions with a few C++ classes that implement popular Image Processing and Computer Vision algorithms.

Computer vision is the science that means to give a comparative, if not better, capacity to a machine or PC. Computer vision is worried about the programmed extraction, investigation and comprehension of valuable data from a single picture or a grouping of pictures. Some of the basic image processing capabilities include filtering, edge detection, corner detection, sampling and interpolation, color conversion, morphological operations, histograms and many more.

Color detection using OpenCV has many advantages like, it allows the detection of a specific color in a livestream video content. In this OpenCV color detection system there are four major modules, activated webcam, scan object, match frame parts and system results. Users can open webcam by clicking the webcam button. Then the algorithm analysis the pattern of the framed part of webcam. Pattern is matched with defined color pattern by RGB color model. If the pattern matched with the potential pattern of RGB color model then the system results with the correct output.

ACKNOWLEDGEMENT

We take this opportunity to express our sincere gratitude to all the people who have helped us to develop our project “**Color Detection using Pandas & OpenCV**” and this report. Their contribution has been of immense use that we find it difficult to acknowledge them in any manner but in writing.

We sincerely acknowledge the guidance of our Principal **Dr. Alam N. Shaikh** Sir which helped us immensely while developing this project.

We would like to express our special thanks of gratitude to our Project Guide and the Head of Department of Computer Engineering, **Prof. (Dr.) Mahavir Devmane** Sir who gave us the golden opportunity to do this wonderful project.

We also like to express a deep sense of gratitude to our project guide **Prof. Sumit Shinde** Sir. For whole hearted support and guidance and being interested during process of our project. Without his inspiration and encouragement, the completion of project would have been difficult task, we would like to recall and thanks all those who have directly and indirectly contributed to its success.

We also express our profound gratitude to Management of Padmabhushan Vasantdada Patil Pratishthan's College of Engineering for giving us the opportunity and facilitates to accomplish our project work. We also express our devoted and sincere thanks to all our teaching and non-teaching staff of Computer Engineering Department who have offered their valuable suggestion and constructive improvisation.

Thankfully,
Shreeram Geedh
Atharva Hanjankar
Selvaganapathi Murugesan

INDEX

CHAPTER 1: INTRODUCTION.....	8
1.1 INTRODUCTION.....	9
1.2 Aim & Objective.....	10
 CHAPTER 2: LITERATURE SURVEY.....	 11
2.1 OVERVIEW OF PYTHON.....	12
2.2 OVERVIEW OF PANDAS PACKAGE IN PYTHON.....	13
2.3 OVERVIEW OF OPENCV	14
 CHAPTER 3: IMPLEMENTATION... ..	 15
3.1 INTRODUCTION	16
3.2 .WHAT IS A PIXEL	16
3.3 ACTUAL WORKING	19
3.4 .OUTPUT.....	23
 CHAPTER 4: SYSTEM ANALYSIS AND SYSTEM DESIGN.....	 24
4.1 ALGORITHM OF PROPOSED SYSTEM	25
4.2 FLOWCHART OF PROPOSED TECHNIQUE	27
4.3 ACTIVITY DIAGRAM.....	28
4.4 USE CASE DIAGRAM.....	29
4.5 SYSTEM ARCHITECTURE.....	30
4.6 HARDWARE REQUIREMENT.....	31

4.7 SOFTWARE REQUIREMENT.....	31
4.8 ADVANTAGE.....	31
4.9 LIMITATION.....	32
4.10 APPLICATION.....	32

CHAPTER 5: FUTURE SCOPE & CONCLUSION..... 33

5.1 FUTURE SCOPE.....	34
5.2 REFERENCE.....	35
5.3 SUMMARY.....	35
5.4 CONCLUSION.....	35

Chapter 1

INTRODUCTION

1.1 Introduction:

Colour detection is the process of detecting the name of any color. Simple isn't it? Well, for humans this is an extremely easy task but for computers, it is not straightforward. Human eyes and brains work together to translate light into color. Light receptors that are present in our eyes transmit the signal to the brain.

Our brain then recognizes the color. Since childhood, we have mapped certain lights with their color names. We will be using the somewhat same strategy to detect color names.

Color Detection In this color detection Python project, we are going to build an application through which you can automatically get the name of the color by clicking on them.

So for this, we will have a data file that contains the color name and its values. Then we will calculate the distance from each color and find the shortest one.

The Dataset

Colours are made up of 3 primary colours; red, green, and blue. In computers, we define each colour value within a range of 0 to 255. So in how many ways we can define a colour? The answer is $256 \times 256 \times 256 = 16,581,375$. There are approximately 16.5 million different ways to represent a color. In our dataset, we need to map each color's values with their corresponding names. But don't worry, we don't need to map all the values. We will be using a dataset

that contains RGB values with their corresponding names. The CSV file for our dataset has been taken from this link:

Colors Dataset

The **colors.csv** file includes 865 color names along with their RGB and hex values.

1.2.Aim & Objective :

In this color detection Python project, we are going to build an application through which you can automatically get the name of the color by clicking on them. So for this, we will have a data file that contains the color name and its values. Then we will calculate the distance from each color and find the shortest one.

To implement a program that can predict any object (image) colour by calculating the RGB value of that pixel which will be selected.

CHAPTER 2

LITERATURE SURVEY

2.1 .Overview Of Python –

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Python is Interpreted – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

Python is Interactive – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python is Object-Oriented – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

Python is a Beginner's Language – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

2.2.Overview Of Pandas Package in Python -

pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, **real world** data analysis in Python. Additionally, it has the broader goal of becoming **the most powerful and flexible open source data analysis / manipulation tool available in any language**. It is already well on its way toward this goal.

pandas is well suited for many different kinds of data:

- Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet
- Ordered and unordered (not necessarily fixed-frequency) time series data.
- Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels
- Any other form of observational / statistical data sets. The data actually need not be labeled at all to be placed into a pandas data structure

2.3.Overview Of OpenCV:

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash.

Computer Vision overlaps significantly with the following fields –

- **Image Processing** – It focuses on image manipulation.
- **Pattern Recognition** – It explains various techniques to classify patterns.
- **Photogrammetry** – It is concerned with obtaining accurate measurements from images.

CHAPTER 3

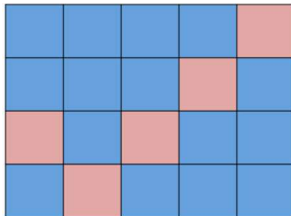
DESIGN & IMPLEMENTATION

3.Introduction :

3.1 What is a Pixel?

Digital images are made of pixels. A *pixel* is a small point of colored light. When you look at a computer monitor, the image you see is actually made of a grid of these tiny dots of light. They are so small and so close together that it looks like one continuous picture. To get an idea of how small a pixel is, the monitor that I happen to be using as I write this has a *resolution* of 1440 x 900 (read as "1440 by 900"). That means that there are 1,440 pixels across the top and 900 pixels down one side, for a total of almost 1.3 million pixels.

This is what pixels might look like if they were magnified. This is an example of a 5x4 image because it is 5 pixels wide and 4 pixels tall.



Each pixel has a color value. We need a way to represent colors so that we can tell the computer which color to make each pixel. There are many color representations, but JavaScript uses a scheme called the *RGBA color model*. Basically, it means that a color is represented by four numbers:

- R (the amount of red light)
- G (the amount of green light)
- B (the amount of blue light) and
- A (called "alpha", this number tells how transparent the color should be)

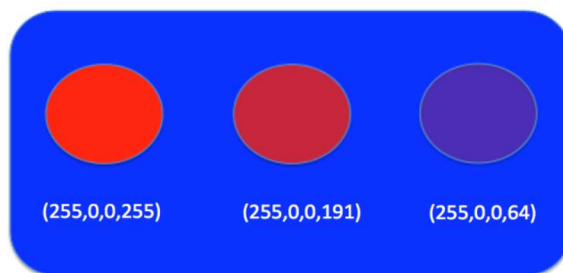
So each color is represented by these four numbers ("Everything's a number", remember?). Moreover, **each of these number slots must have a value between 0 and 255**. You may be wondering whether or not only 256 possibilities for each slot is enough to make all the many colors that we might want. If you have 256 possibilities for each of the R, G, and B values, (ignoring the transparency number for now) then the



total number of colors available is over 16 million! It is estimated that the human eye can only detect 10 million different colors, so there's no worry that you won't be able to make any color you want.

Since the computer uses light to make a picture, the RGBA model is an *additive color model*. This means that the more the medium is added, the closer the color gets to white. Contrast this with using paint as a medium. That is a *subtractive color model* because the more paint you add, the further you get from white. So it should come as no surprise then that a color with R, G, and B values all 0 is black (no light) and a color with R, G, B values all 255 is white (pure light). Think of these numbers as knobs that you can turn up or down. If you turn on the red and blue lights and leave off the green light, you have shades of purple (R = 150, G = 0, B = 150, for example). The more you turn up the light, the higher the number goes and the brighter the color gets. How will you know what values to use for R, G, and B to make the color you want? If you search for "RGB color chart" on the Internet you will find lots of sites with palettes of colors and their corresponding values.

Transparency: Alpha Channel



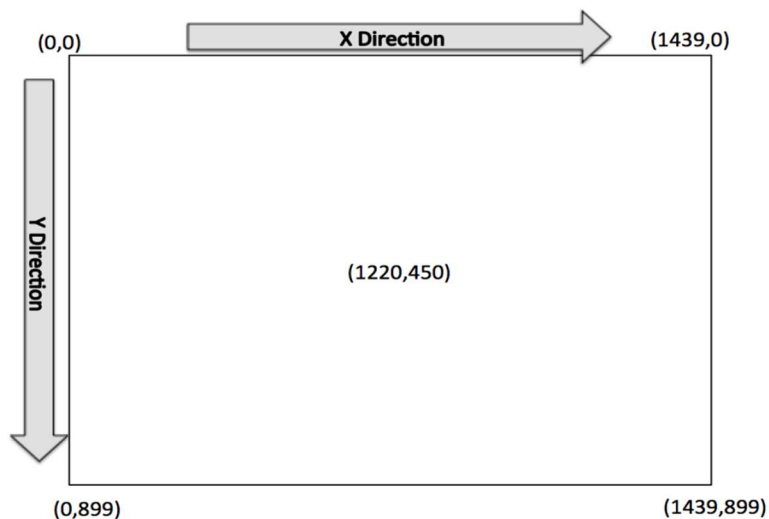
The last value, called alpha, is also a number whose value must be between 0 and 255. This time the value of the number does not change the color's hue, but rather the transparency of the color. If a pixel has an alpha value of 0, it is completely transparent, or invisible. If it has a value of 255, it is completely opaque. Using this transparency value allows you to have layers of color on the screen, or shapes that can be partially seen through other shapes.

Image Coordinate System

Finally, it is important to be able to distinguish one pixel from another. We do this by referring to each pixel's location on the screen or image. Each pixel gets

an *X and Y value*, where (0,0) is the top left corner of the screen. (This can take some getting used to as it's not the way a Cartesian Plane is drawn in math class!)

The X value refers to how far right the pixel is, and the Y value refers to how far down the pixel is. For my computer monitor, which is 1440 x 900, the top left corner is (0,0), the top right is (1439,0), the bottom left is (0,899), the bottom right is (1439,899), and the middle is (720, 450).



All of the discussion so far has been generic knowledge of a computer representation of images and colors. In other words, the same information would probably apply if you were programming in a language other than JavaScript. The rest of this page gives you details that are specific to JavaScript and the problems you are asked to solve.

3.2. Actual Working :

1. Taking an image from the user

We are using argparse library to create an argument parser. We can directly give an image path from the command prompt:

```
1 import argparse
2 ap = argparse.ArgumentParser()
3 ap.add_argument('-i', '--image', required=True, help="Image Path")
4 args = vars(ap.parse_args())
5 img_path = args['image']
6 #Reading image with opencv
7 img = cv2.imread(img_path)
```

2. Next, we read the CSV file with pandas

The pandas library is very useful when we need to perform various operations on data files like CSV. **pd.read_csv()** reads the CSV file and loads it into the pandas DataFrame. We have assigned each column with a name for easy accessing.

```
1 #Reading csv file with pandas and giving names to each column
2 index=["color", "color_name", "hex", "R", "G", "B"]
3 csv = pd.read_csv('colors.csv', names=index, header=None)
```

3. Set a mouse callback event on a window

First, we created a window in which the input image will display. Then, we set a callback function which will be called when a mouse event happens.

```
1 cv2.namedWindow('image')
2 cv2.setMouseCallback('image',draw_function)
```

4. Create the draw_function

It will calculate the rgb values of the pixel which we double click. The function parameters have the event name, (x,y) coordinates of the mouse position, etc. In the function, we check if the event is double-clicked then we calculate and set the r,g,b values along with x,y positions of the mouse.

```
1 def draw_function(event, x,y,flags,param):
2     if event == cv2.EVENT_LBUTTONDBLCLK:
3         global b,g,r,xpos,ypos, clicked
4         clicked = True
5         xpos = x
6         ypos = y
7         b,g,r = img[y,x]
8         b = int(b)
9         g = int(g)
10        r = int(r)
```

5. Calculate distance to get color name

We have the r,g and b values. Now, we need another function which will return us the color name from RGB values. To get the color name, we calculate a distance(d) which tells us how

close we are to color and choose the one having minimum distance.

Our distance is calculated by this formula:

$$d = \text{abs}(\text{Red} - \text{ithRedColor}) + (\text{Green} - \text{ithGreenColor}) + (\text{Blue} - \text{ithBlueColor})$$

```
1 def getColorName(R,G,B):
2     minimum = 10000
3     for i in range(len(csv)):
4         d = abs(R- int(csv.loc[i,"R"])) + abs(G- int(csv.loc[i,"G"]))+ abs(B- int(csv.loc[i,"B"]))
5         if(d<=minimum):
6             minimum = d
7             cname = csv.loc[i,"color_name"]
8     return cname
```

6. Display image on the window

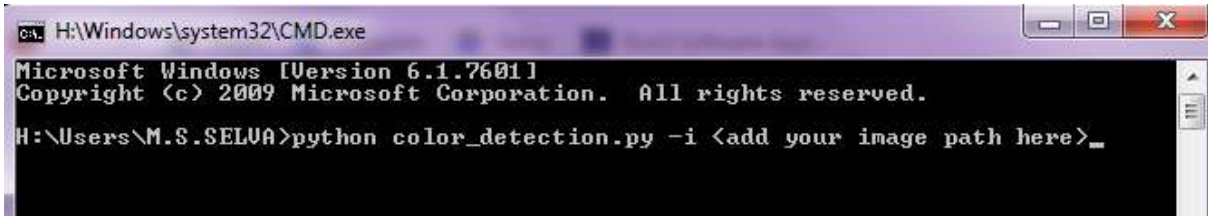
Whenever a double click event occurs, it will update the color name and RGB values on the window.

Using the **cv2.imshow()** function, we draw the image on the window. When the user double clicks the window, we draw a rectangle and get the color name to draw text on the window using **cv2.rectangle** and **cv2.putText()** functions.

```
1 while(1):
2     cv2.imshow("image",img)
3     if (clicked):
4         #cv2.rectangle(image, startpoint, endpoint, color, thickness) -1 thickness fills rectangle entirely
5         cv2.rectangle(img,(20,20), (750,60), (b,g,r), -1)
6         #Creating text string to display ( Color name and RGB values )
7         text = getColorName(r,g,b) + ' R='+ str(r) + ' G='+ str(g) + ' B='+ str(b)
8         #cv2.putText(img,text,start,font(0-7), fontScale, color, thickness, lineType, (optional bottomLeft bool) )
9         cv2.putText(img, text,(50,50),2,0.8,(255,255,255),2,cv2.LINE_AA)
10        #For very light colours we will display text in black colour
11        if(r+g+b>600):
12            cv2.putText(img, text,(50,50),2,0.8,(0,0,0),2,cv2.LINE_AA)
13        clicked=False
14        #Break the loop when user hits 'esc' key
15        if cv2.waitKey(20) & 0xFF ==27:
16            break
17    cv2.destroyAllWindows()
```

8. Run Python File

The beginner Python project is now complete, you can run the Python file from the command prompt. Make sure to give an image path using '-i' argument. If the image is in another directory, then you need to give full path of the image:

A screenshot of a Windows Command Prompt window. The title bar at the top reads "H:\Windows\system32\CMD.exe". The window content shows the following text: "Microsoft Windows [Version 6.1.7601] Copyright (c) 2009 Microsoft Corporation. All rights reserved. H:\Users\M.S.SELVA>python color_detection.py -i <add your image path here>_". The prompt is followed by an underscore character, indicating the command is ready to be executed.

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

H:\Users\M.S.SELVA>python color_detection.py -i <add your image path here>_
```


3.3.Output:-



CHAPTER 4

SYSTEM ANALYSIS & SYSTEM DESIGN

4.1. Algorithm of Proposed Technique

The input image is in RGB color space from camera. Each input image is partitioned into blocks as shown in Figure 4.4. The block size is of 4×4 pixels. There is one centroid value and one flag value in each block. The centroid value is calculated by averaging the color value of the block. By partitioning the input image in 4×4 blocks, the resultant image is of $\frac{1}{4}$ resolution of an input image. The entire process is shown in below Figure 4.4.

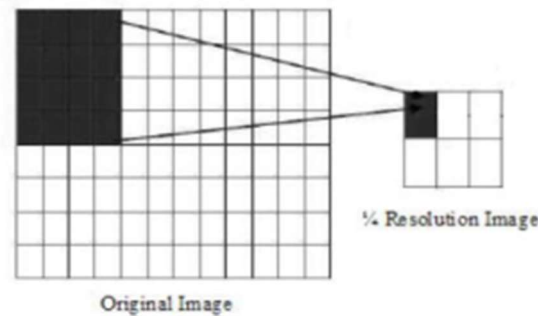


Figure 4.4: Generating $\frac{1}{4}$ Resolution Image

Although there are many color spaces, in this work $Y C_r C_b$ color space is opted because its effectiveness in skin detection has been shown previously in literature [135]-[138].

$Y C_r C_b$, the RGB components are separated into luminance (Y), chrominance blue (C_b) and chrominance red (C_r). The standard relation between $Y C_r C_b$ and RGB color space is

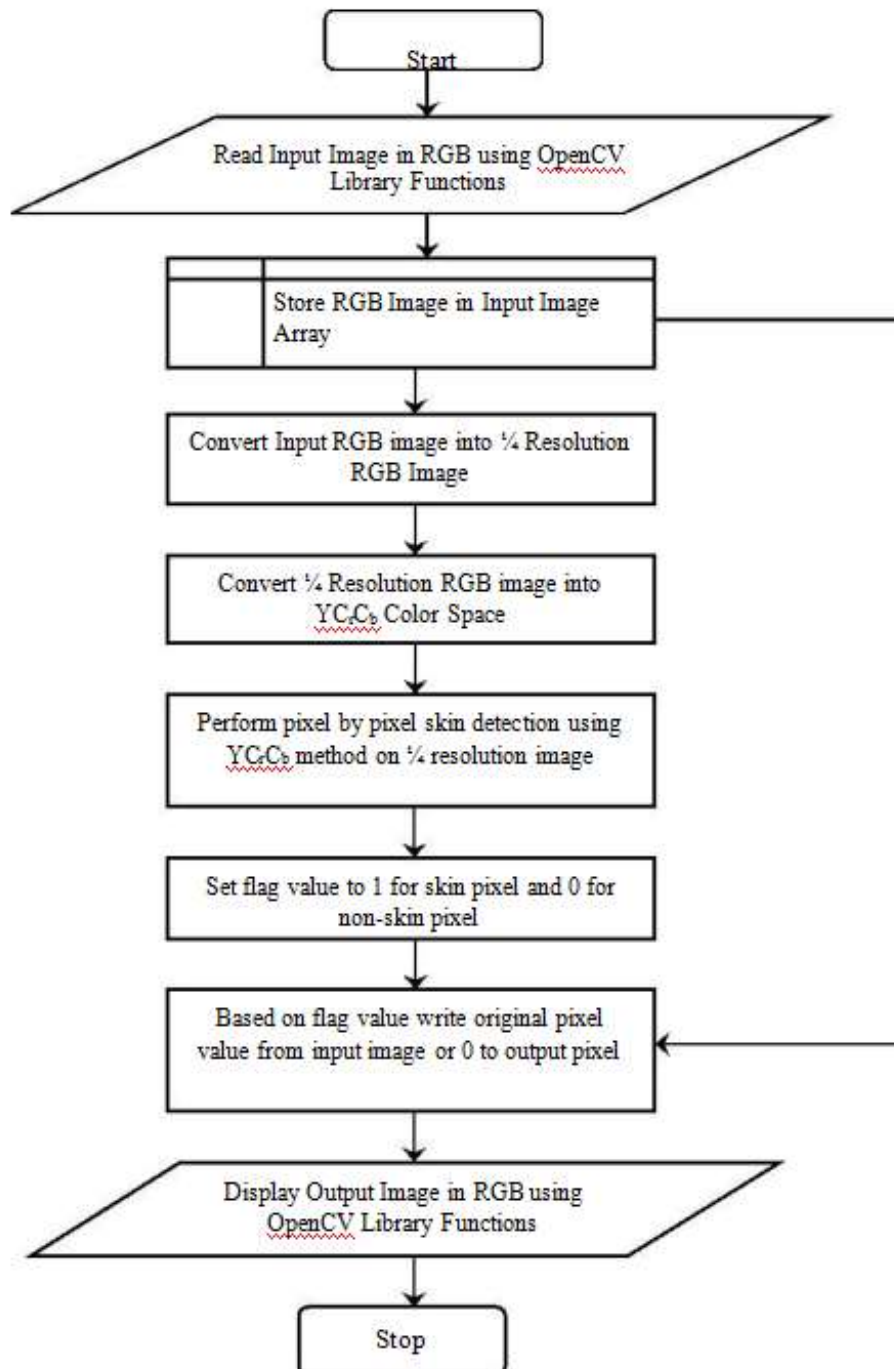
$$Y > 80; 85 < C_b < 135; 135 < C_r < 180$$

From RGB to $YCrCb$ color space transformation is performed on the $\frac{1}{4}$ resolution image. This saves 16 times the computational resources required for color space transformation on original image. The skin color detections conditions for $YCrCb$ color space [115], [139]-[141] are given below:

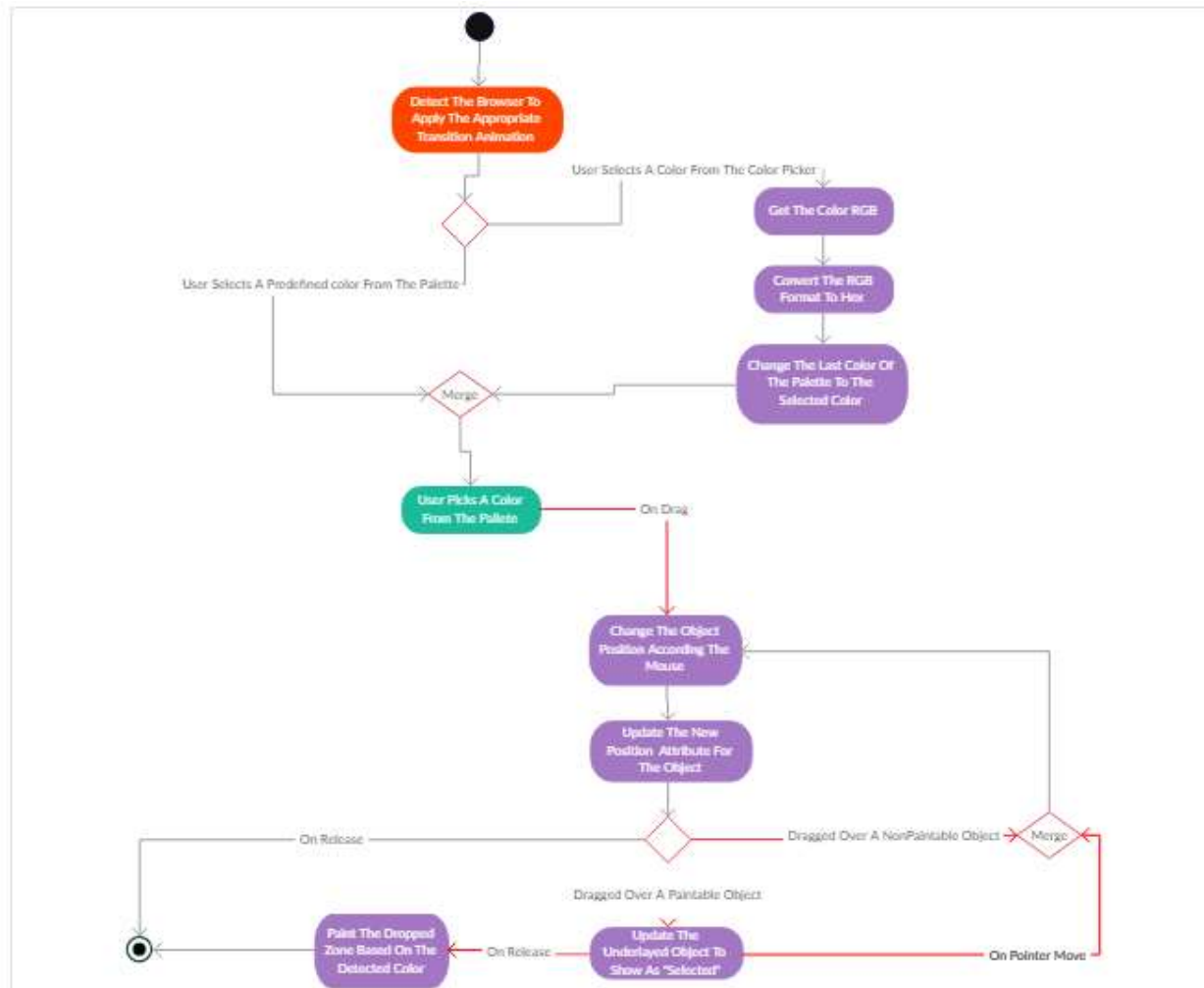
The skin color detection is performed on this $\frac{1}{4}$ resolution image using $YCrCb$ method. The corresponding flag value is set to 1 or 0 based on skin pixel or non-skin pixel in $\frac{1}{4}$ resolution image. The major steps involved in the algorithm are as follow:

- Each RGB image is partitioned into 4x4 blocks.
- For each block the centroid value is computed by averaging color value.
- Result is an image of $\frac{1}{4}$ resolution of input image.
- Skin color detection is performed for this resulted $\frac{1}{4}$ resolution image using $YCrCb$ Method.
- Set the flag value for each skin pixel as 1.
- Set the flag value for each non-skin pixel as 0.
- If flag value is 1 then write the original pixels from input image to output image for that block.
- If flag value is 0 then write black color i.e. 0 for all pixels of the corresponding block in output image.

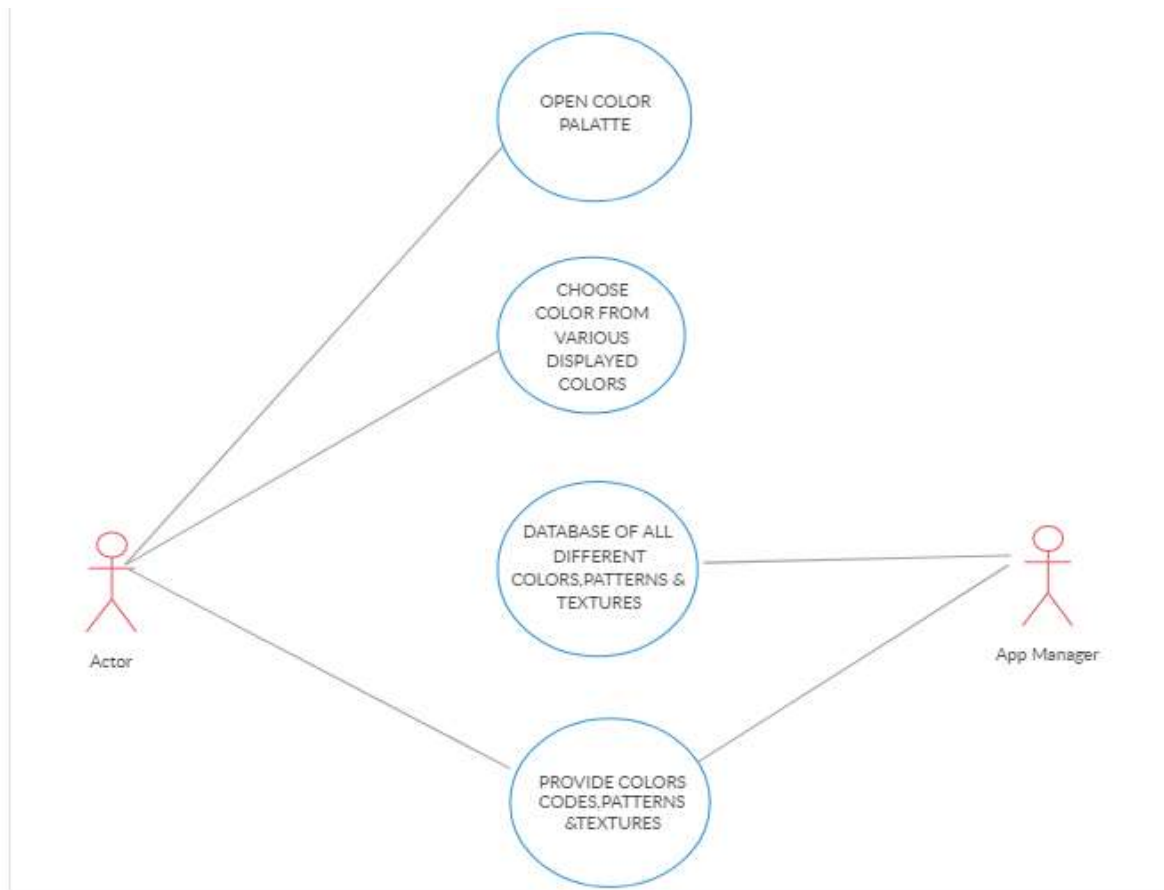
4.2 FLOWCHART OF PROPOSED SYSTEM



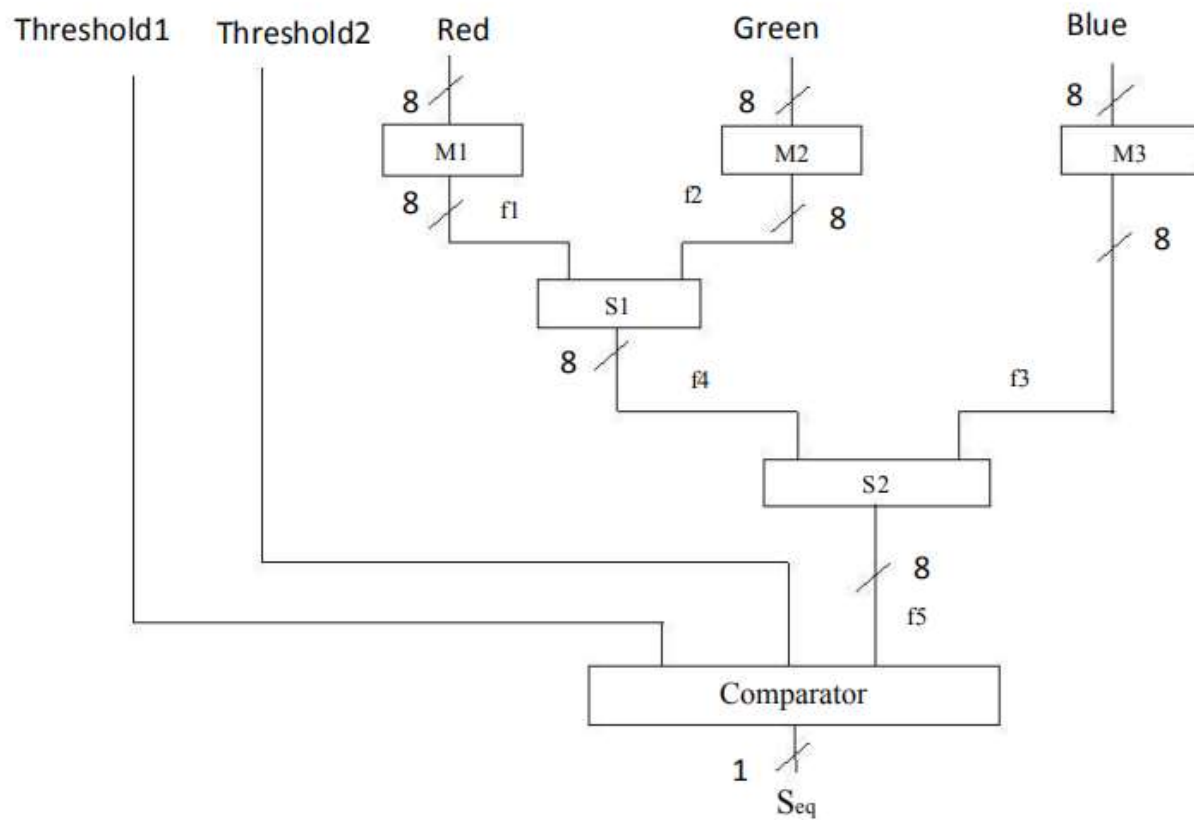
4.3.ACTIVITY DIAGRAM



4.4.USECASE DIAGRAM



4.5.SYSTEM ARCHITECTURE



4.6.Hardware Requirement:

- i3 Processor Based Computer or higher
- Memory: 1 GB RAM
- Hard Drive: 50 GB
- Monitor
- Internet Connection

4.7.Software Requirement:

- Windows 7 or higher
- Python
- Django
- MySQL database

4.8.Advantages

- The system is user-friendly and has simple interface.
- Can be used in manufacturing company

4.9.Limitation

- Data need to be entered properly otherwise, outcome may won't be accurate.

4.10.Application

- This system can be used by the multiple peoples to get the counselling sessions online.

CHAPTER 5

FUTURE SCOPE &

CONCLUSION

5.1 FUTURE SCOPE :

1. Computer vision- Color detection is the basic and important step for proceeding in computer vision. Some special type of spectacles can be made which will make use of computer vision (image processing) along with neural networks to provide an artificial vision to blind people.

2. Spy robots- The spy robots are made to identify objects in the place where they are launched. Object's shape, size, color, orientation is of importance to robot.

3. Object Segregation- An object can be segregated (separated) on the basis of its color.

4. Object Tracking- A moving object can be tracked on the basis of its coordinate axis.

5.2. Reference

- ✓ <https://shsu-ir.tdl.org/shsu-ir/bitstream/handle/20.500.11875/1164/0781.pdf?sequence=1>
- ✓ <https://ieeexplore.ieee.org/document/6208293/>
- ✓ <https://ieeexplore.ieee.org/document/4679917/>

5.3. Summary

In this Python project with source code, we learned about colors and how we can extract color RGB values and the color name of a pixel. We learned how to handle events like double-clicking on the window and saw how to read CSV files with pandas and perform operations on data. This is used in numerous image editing and drawing apps.

5.4. Conclusion: -

From this project we conclude that color plays a vitally important role in the world in which we live. Color can sway thinking, change actions, and cause reactions. Color tends to be the most important part

in object color detection. Colors can be expressed in many ways and It is the only way that distinct two objects for a computer.

Color detection using pandas & opencv boosts up yhe accuracy of algorithms we also came to know that color can be helpful in system automation and can have numerous applications in near future.